

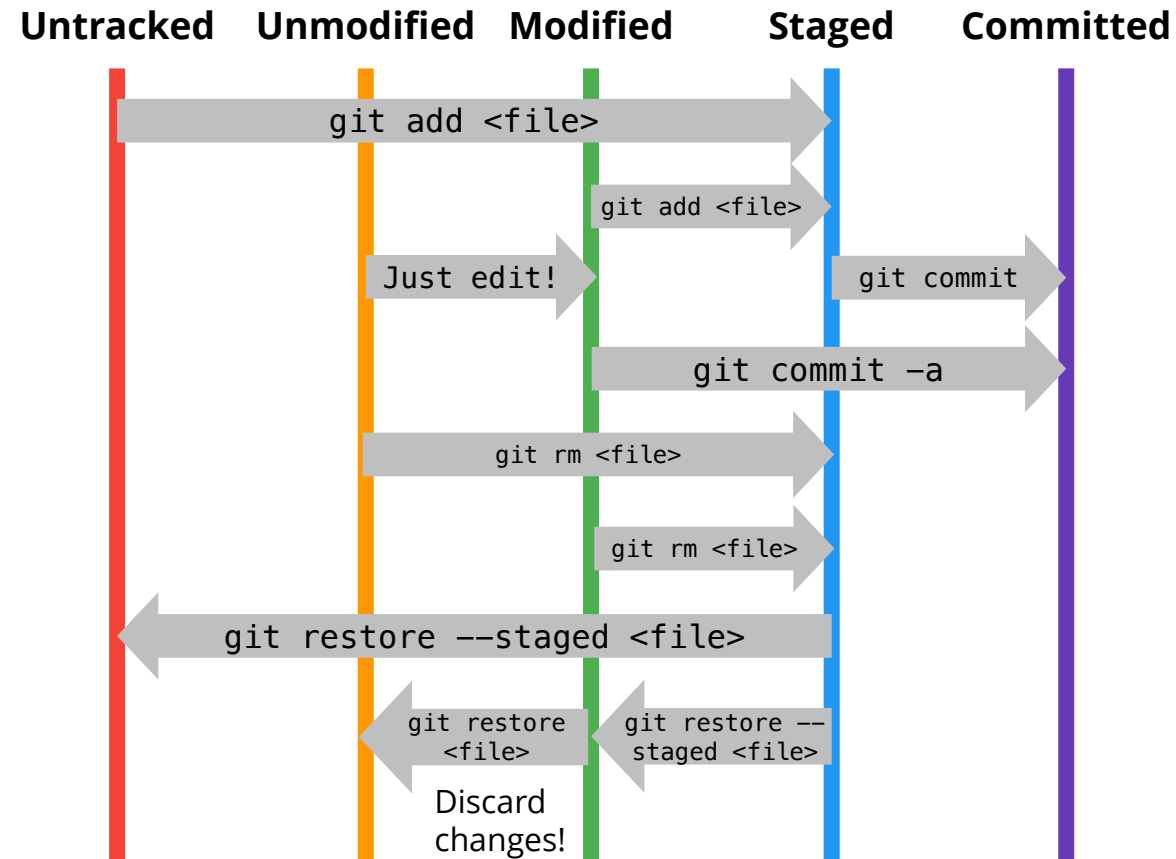
# Open-Source Software Practice

## Lab 03. Git Advanced

Instructor: Jaemin Jo (조재민, [jmjo@skku.edu](mailto:jmjo@skku.edu))

Interactive Data Computing Lab (*IDCLab*),  
College of Computing and Informatics,  
Sungkyunkwan University

# Review: Git Basics



# Goals

---

1. Branching
2. Merging
3. Resolving conflicts
4. Sign up for GitHub
5. Create a repository on GitHub
6. Working with remotes

# Let's Practice – 1 (init)

---

1. `git init`
2. `echo "Initial README" >> README.md`
3. `git add README.md`
4. `git commit`

# Let's Practice – 2 (creating a branch)

1. git branch
2. git branch dev
3. git branch
4. git checkout dev
5. git branch

```
MINGW64:/d/test

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test (main)
$ git branch
* main

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test (main)
$ git branch dev

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test (main)
$ git branch
  dev
* main

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test (main)
$ git checkout dev
Switched to branch 'dev'

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test (dev)
$ git branch
* dev
  main

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test (dev)
$ |
```

# Let's Practice – 3 (commit)

- You are on the *dev* branch.

1. `echo "Change on dev" >> README.md`
2. `git add .`
3. `git commit -m "Changed README.md"`
4. `git log --graph --all --decorate --oneline`

```
MINGW64:/d/test3

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test3 (dev)
$ git log --graph --all --decorate --oneline
* 2d48dba (HEAD -> dev) Changed README.md
* 9650c09 (main) Added README.md

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test3 (dev)
$ |
```

# Let's Practice – 4 (going back to main)

- You are on the *dev* branch.
1. `git checkout main`
  2. Open README.md and check out that the last change you made (on the *dev* branch) was reverted.
  3. `echo "Change on main" >> README.md`
  4. `git add .`
  5. `git commit -m "Changed README.md for the second time"`
  6. `git log --graph --all --decorate --oneline`

# Let's Practice - 4

```
MINGW64:/d/test3

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test3 (main)
$ git log --graph --all --decorate --oneline
* 6f73f38 (HEAD -> main) Changed README.md for the second time
| * 2d48dba (dev) Changed README.md
|/
* 9650c09 Added README.md

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test3 (main)
$
```

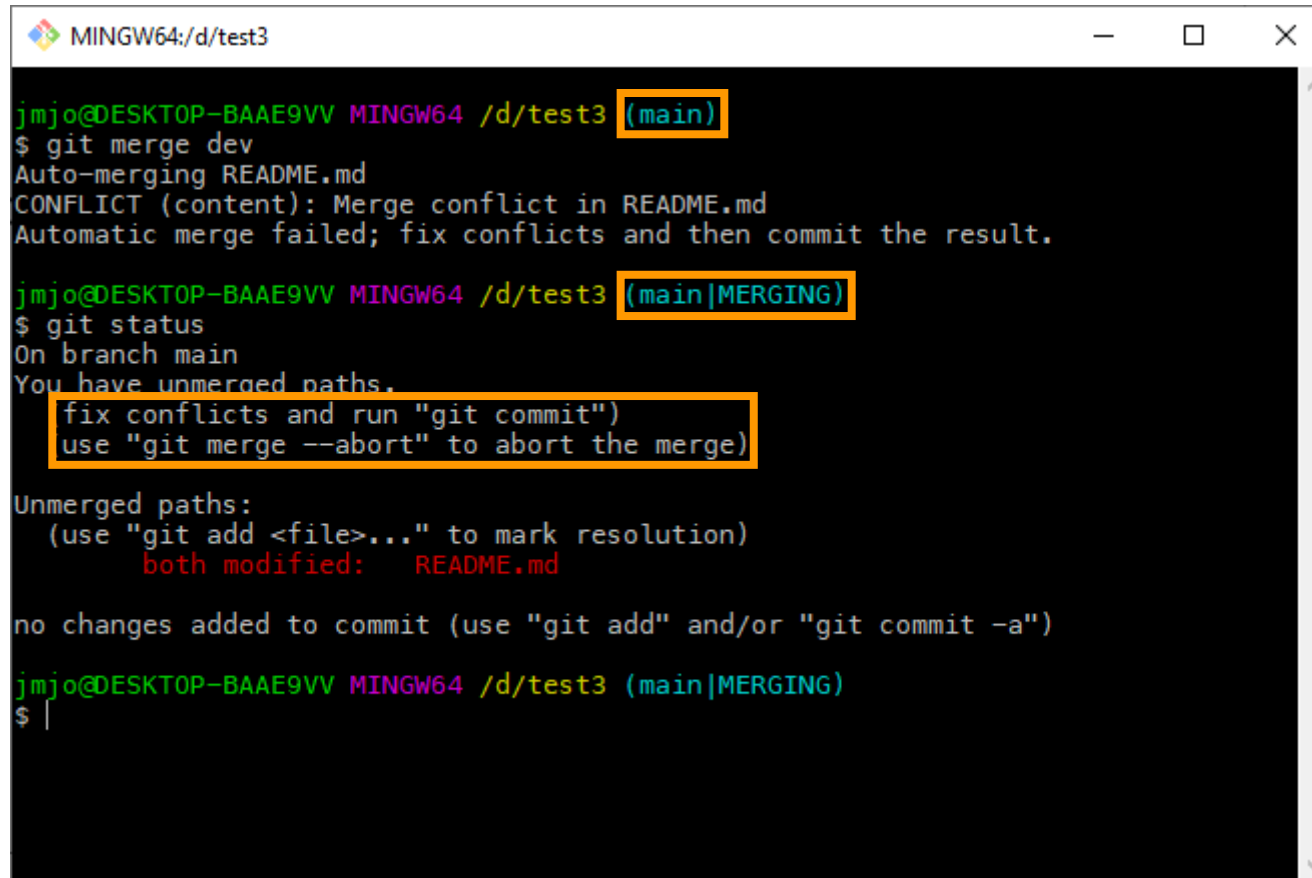


# Let's Practice – 5 (merging)

- You should be on the *main* branch. Otherwise, `git checkout main`
1. `git merge dev`
    - The *dev* branch does not change. We only change the branch we are currently on, *main*.
  2. Check out the error message.
  3. `git status`

# Let's Practice – 5 (merging)

- Check out the hints from Git.



```
MINGW64:/d/test3
jmjo@DESKTOP-BAAE9VV MINGW64 /d/test3 (main)
$ git merge dev
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.

jmjo@DESKTOP-BAAE9VV MINGW64 /d/test3 (main|MERGING)
$ git status
On branch main
You have unmerged paths.
  fix conflicts and run "git commit"
  use "git merge --abort" to abort the merge)

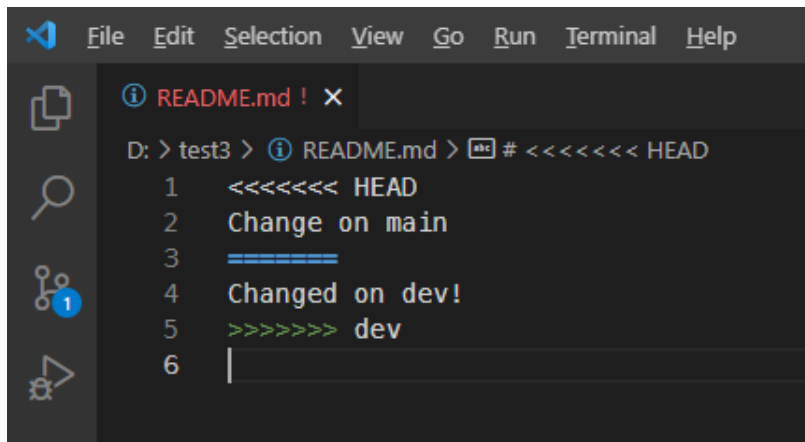
Unmerged paths:
  (use "git add <file>..." to mark resolution)
    both modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

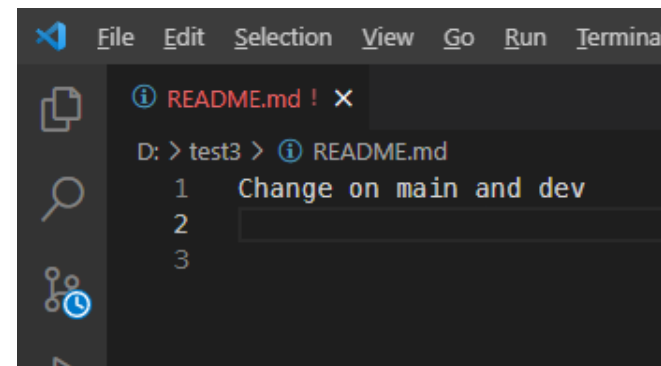
jmjo@DESKTOP-BAAE9VV MINGW64 /d/test3 (main|MERGING)
$ |
```

# Let's Practice – 6 (resolving conflicts)

1. Open *README.md* and edit
2. `git add .`
3. `git status`
4. `git commit -m "Merged dev"`
5. `git log --graph --all --decorate --oneline`



```
File Edit Selection View Go Run Terminal Help
D: > test3 > README.md > # <<<<<<< HEAD
1 <<<<<<< HEAD
2 Change on main
3 =====
4 Changed on dev!
5 >>>>>>> dev
6 |
```



```
File Edit Selection View Go Run Terminal
D: > test3 > README.md
1 Change on main and dev
2
3
```

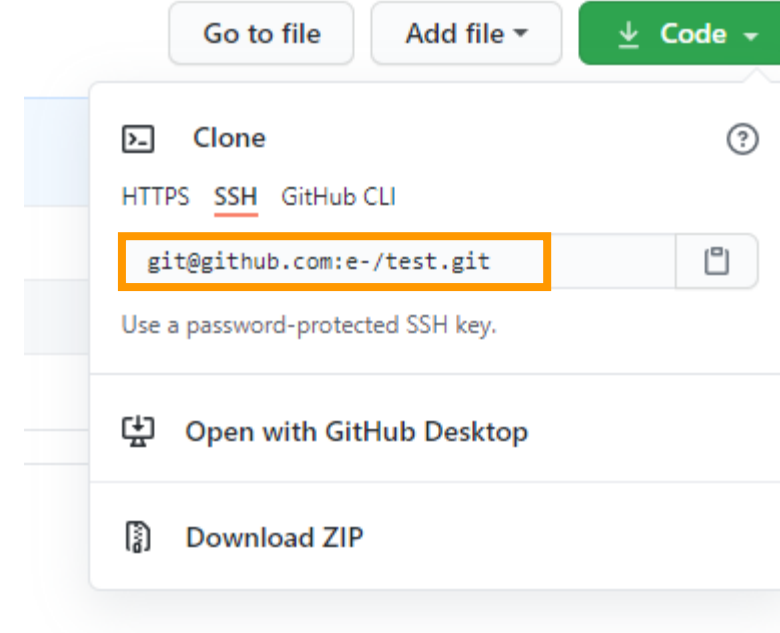
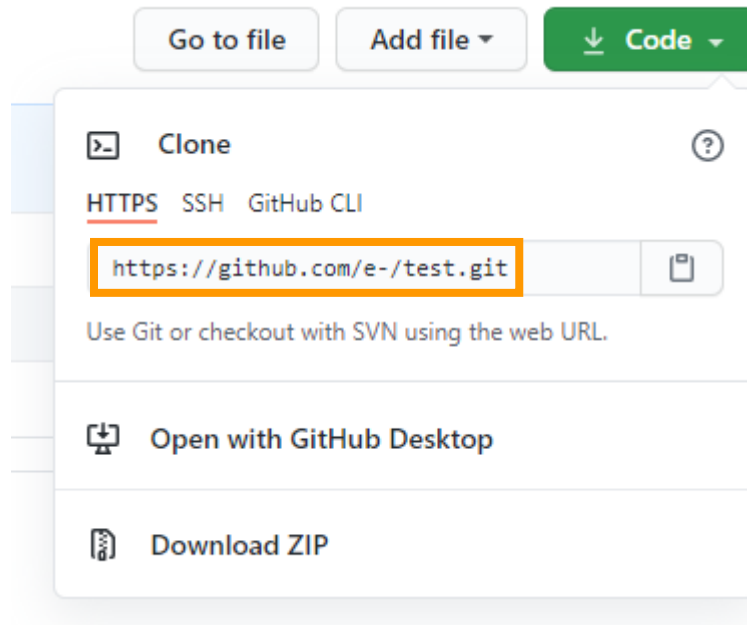
# Let's Practice – 6 (resolving conflicts)

```
MINGW64:/d/test3

jmo@DESKTOP-BAAE9VV MINGW64 /d/test3 (main)
$ git log --graph --all --decorate --oneline
* 2022f7c (HEAD -> main) Merged dev
| \
| * 2d48dba (dev) Changed README.md
* | 6f73f38 Changed README.md for the second time
|/
* 9650c09 Added README.md

jmo@DESKTOP-BAAE9VV MINGW64 /d/test3 (main)
$
```

# Cloning a Remote Repo



# Cloning a Remote Repo

- You just want to see others' repo (i.e., read-only)
  - `git clone https://github.com/...`
  - The most convenient way
  - Requires log-in to push
- You want to change your repo
  - `git clone git@github.com:...`
  - Requires key registration to push
- You want to change others' repo (covered in the next class)
  - Be invited to the repo as a collaborator (rare), or
  - Fork the repo and clone *your* version of the repo.

# Let's Practice – 7 (cloning twbs)

- Navigate to a new directory. Make sure that you are not in a Git repo.
1. `git clone https://github.com/twbs/bootstrap`
  2. `git branch --remote`
  3. `git log --all --decorate --online --graph`
    - Page down or up to navigate, press 'q' to quit
  4. Visit <https://github.com/twbs/bootstrap> to check the branch names and commits.

# Let's Practice – 7 (cloning)

1. Sign in to GitHub
2. Create a new repository
3. Clone the repo by `git clone git@github.com:...`
4. Error!

```
git@github.com: Permission denied (publickey).  
fatal: Could not read from remote repository.
```

Please make sure you have the correct access rights  
and the repository exists.


A repository contains all project files, including the revision history. Already have a project repository elsewhere?  
[Import a repository.](#)


## Create a new repository

Owner \* / Repository name \*

Great repository names are short and memorable. Need inspiration? How about [probable-bassoon](#)?

Description (optional)

☒  **Public**  
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**  
You choose who can see and commit to this repository.

**Initialize this repository with:**  
Skip this step if you're importing an existing repository.

☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more.](#)

☒ **Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)

☒ **Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)

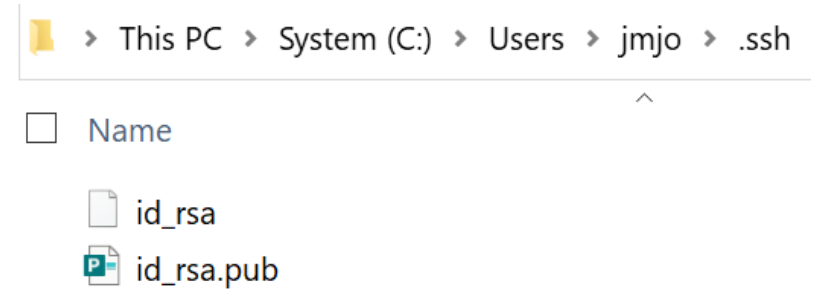
This will set `main` as the default branch. Change the default name in your [settings](#).

[Create repository](#)



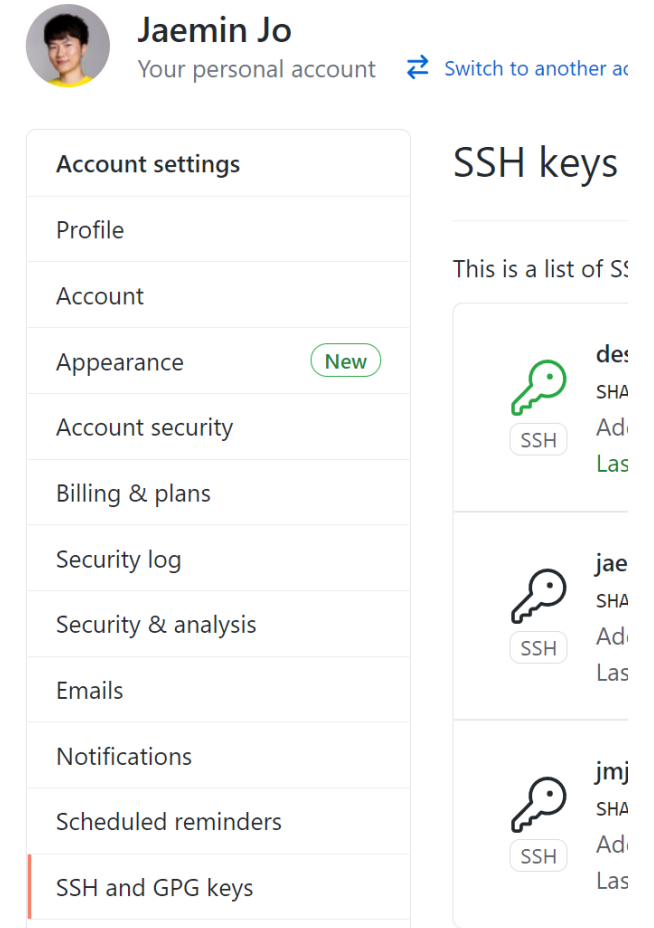
# Let's Practice – 8 (ssh-keygen)

1. Make sure keys do not exist: `ls ~/.ssh`
2. `ssh-keygen`
3. Use the default path
4. Enter the passphrase
  - A special password to use the keys
5. `cat ~/.ssh/id_rsa.pub`
6. Copy the public key and register it to GitHub



# Let's Practice – 9 (registering keys)

1. Go to SSH and GPG keys tab in Settings.
2. Enter your password again
3. Click on “New SSH Key”
4. Copy the content of “**id\_rsa.pub**” to the form.
  - **NOT** “id\_rsa”. This is your private key.



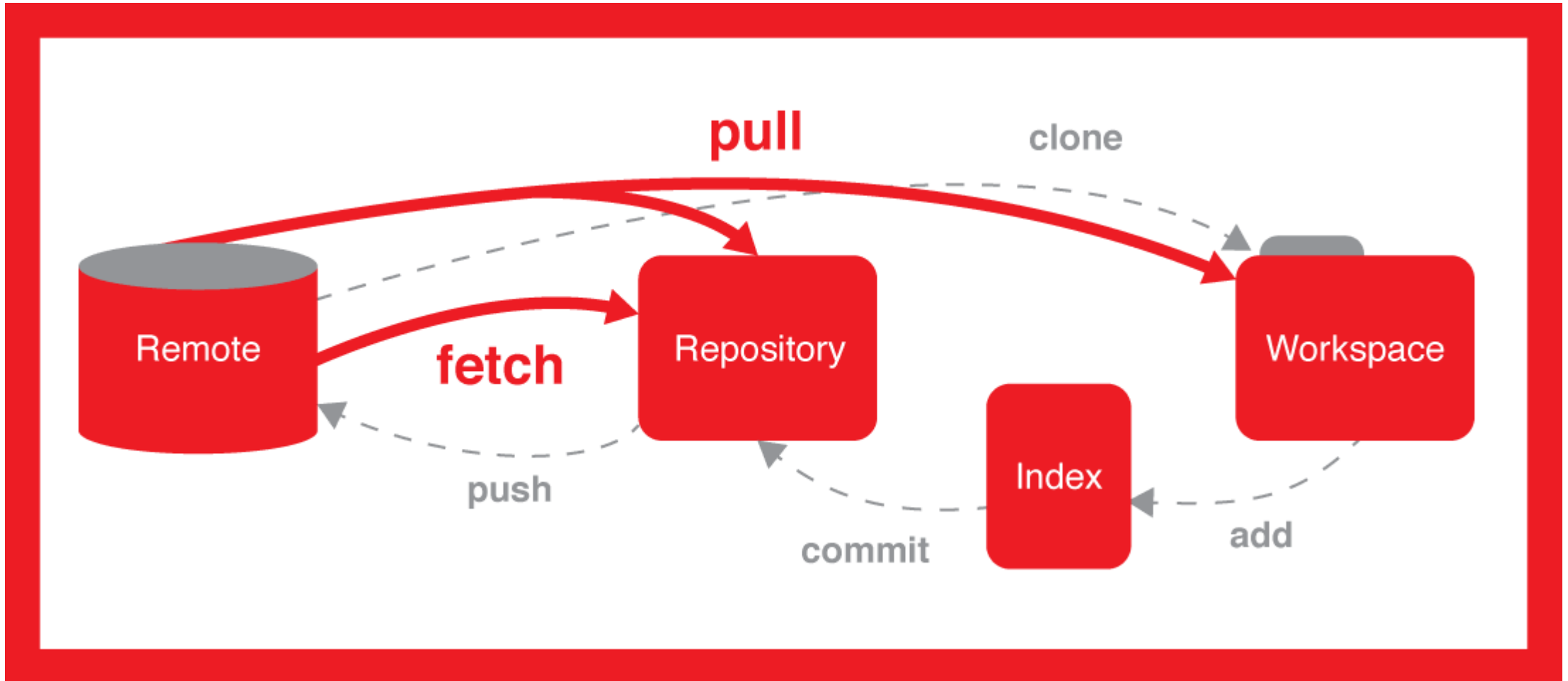
# Let's Practice – 10 (pushing)

1. Initialize a Git repo: `git init`
  - If you already have one, skip this one.
2. Create a README.md file: `code README.md`
  - If you already have one, skip this one.
3. `git add .`
4. `git commit -m "Added README.md"`
5. `git remote add origin git@github.com:e-/test.git`
6. `git push --set-upstream origin main`

# Let's Practice – 10 (pushing)




- `git remote add origin <url>`
  - Register `<url>` as a remote repository with a name “origin”.
  - You can check out the remotes with `git remote -v`
- `git push --set-upstream origin main`
  - Set *origin* as the upstream and push the local repo to the remote.
  - Once the upstream is set, you can just `git push` to upload the changes.

# Workflow






<https://stackoverflow.com/questions/1783405/how-do-i-check-out-a-remote-git-branch>

# Check out the Result


 main  1 branch  0 tags


[Go to file](#) [Add file](#) [Code](#)

 e- Added README.md e1528b6 7 minutes ago  1 commit

 README.md Added README.md 7 minutes ago

Add a README with an overview of your project. [Add a README](#)

 main

 Commits on Sep 11, 2022

Added README.md  e1528b6 

 e- committed 7 minutes ago

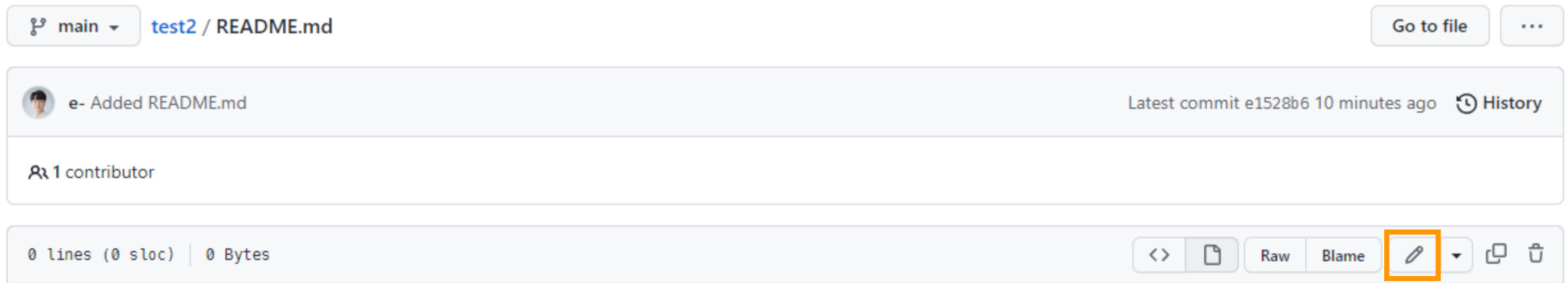
# Let's Practice – 11 (more pushing)

---

1. Add or change files
2. `git add .`
3. `git commit`
4. `git push`
  - You can omit the `--set-upstream` part since it is already set.

# Let's Practice – 12 (pulling)

- Suppose another person updated the remote repo, making your local repo outdated.
- To simulate this, let's change README.md on GitHub.





&lt;&gt; Edit file

Preview

Spaces

2

No wrap

1 Updated via GitHub

**Commit changes**

Update README.md

Add an optional extended description...

# Let's Practice – 12 (pulling)

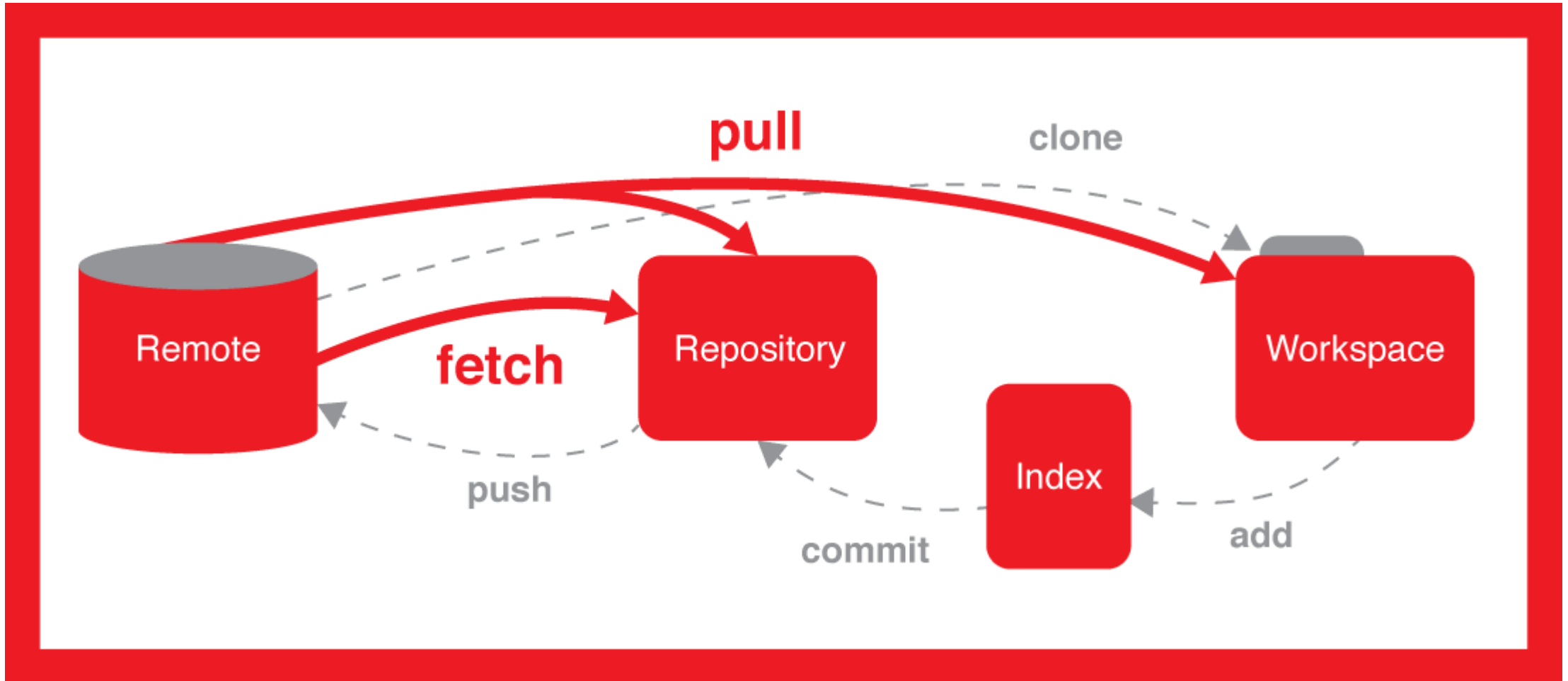
- Now, README.md in the remote is newer than one in the local.
- We need to pull the changes before updating the local version.
- `git pull`

```
MINGW64:/d/ossdp/test2

jmjo@DESKTOP-BAAE9VV MINGW64 /d/ossdp/test2 (main)
$ git pull
Enter passphrase for key '/c/Users/jmjo/.ssh/id_rsa':
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 632 bytes | 79.00 KiB/s, done.
From github.com:e-/test2
   e1528b6..f92282b  main      -> origin/main
Updating e1528b6..f92282b
Fast-forward
 README.md | 1 +
 1 file changed, 1 insertion(+)

jmjo@DESKTOP-BAAE9VV MINGW64 /d/ossdp/test2 (main)
$ |
```

# Workflow



# Pulling the Changes

---

- So, before changing the local code, always `git pull`.
- What happen if you change a file without or before pulling the new version?
- Let's test this.

# Let's Practice – 13 (merge conflicts)

1. Update the remote README.md again via GitHub.
2. Change the local README.md and commit the change.
3. `git push`



```
MINGW64:/d/ossptest2
jmjo@DESKTOP-BAAE9VV MINGW64 /d/ossptest2 (main)
$ git push
Enter passphrase for key '/c/Users/jmjo/.ssh/id_rsa':
To github.com:e-/test2.git
 ! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'github.com:e-/test2.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.

jmjo@DESKTOP-BAAE9VV MINGW64 /d/ossptest2 (main)
$
```

# Let's Practice – 13 (merge conflicts)

- The error message means that someone else has changed the remote while you are working, so you need to sync the changes first before pushing to the remote.
- `git pull`



```
MINGW64:/d/ossptest2
jmjo@DESKTOP-BAAE9VV MINGW64 /d/ossptest2 (main)
$ git push
Enter passphrase for key '/c/Users/jmjo/.ssh/id_rsa':
To github.com:e-/test2.git
 ! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'github.com:e-/test2.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.

jmjo@DESKTOP-BAAE9VV MINGW64 /d/ossptest2 (main)
$
```

# Let's Practice – 14 (resolving conflicts)

- You pulled the changes, but merge conflict happened.
- This is similar to merge conflict between branches.
- code README.md

```
MINGW64:/d/ossptest2

jmo@DESKTOP-BAAE9VV MINGW64 /d/ossptest2 (main)
$ git pull
Enter passphrase for key '/c/Users/jmo/.ssh/id_rsa':
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 652 bytes | 81.00 KiB/s, done.
From github.com:e-/test2
   f92282b..292a707  main       -> origin/main
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.

jmo@DESKTOP-BAAE9VV MINGW64 /d/ossptest2 (main|MERGING)
$
```

# Let's Practice – 14 (resolving conflicts)

- Remove the conflict messages and resolve the conflict.
- `git add .`
- `git commit`
  - The default merging message is already there.

```
❗ README.md ! ●
D: > ossp > test2 > ❗ README.md > [abc] # <<<<<<< HEAD
1 <<<<<<< HEAD
2 Updated locally
3 =====
4 Updated via GitHub for the second time.
5 >>>>>>> 292a707f2857c88496a8bf4e1a5ec482a17633cd
6
```

```
❖ COMMIT_EDITMSG ✕
D: > ossp > test2 > .git > ❖ COMMIT_EDITMSG
1 Merge branch 'main' of github.com:e-/test2
2
3 # Conflicts:
4 #   README.md
5 #
```

```
❗ README.md ! ✕
D: > ossp > test2 > ❗ README.md
1 Updated locally and via GitHub
2
```



# Let's Practice – 14 (resolving conflicts)

- git push

```
MINGW64:/d/ossdp/test2

jmjo@DESKTOP-BAAE9VV MINGW64 /d/ossdp/test2 (main)
$ git push
Enter passphrase for key '/c/Users/jmjo/.ssh/id_rsa':
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 32 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (6/6), 548 bytes | 548.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:e-/test2.git
   292a707..204ae64  main -> main

jmjo@DESKTOP-BAAE9VV MINGW64 /d/ossdp/test2 (main)
$ |
```

# Pulling before Pushing

---

- Always remember that others can change the remote until the moment that you push. So, pull frequently.
- Don't be afraid of merge conflicts. Search for "<<<" and merge the contents between "=====

# Questions..

---

- Can you make branches and push them? How can you check out the branch from the GitHub interface?
- Previously, we learned how to deal with modify/modify conflict? Can you make up a scenario for delete/modify conflicts?